HIGH FREQUENCY COMMUNICATION SYSTEMS

Lecture 6

Hasan T Abbas & Qammer H Abbasi

Spring 2022

LECTURE OUTLINE

- Numerical Methods in Electromagnetics
- Discretisation of Maxwell's Equations
- Finite Difference Time Domain method

ELECTROMAGNETIC SIMULATION METHODS

- There are many numerical methods developed to solve Maxwell's or wave equations

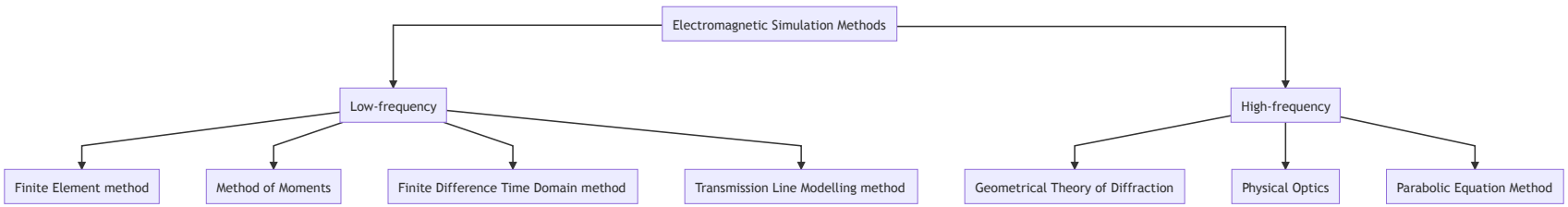


Figure 1: Various simulation methods in EM

- We solve either a differential or integral equation *numerically*
  - Only at a single frequency
- These methods are relatively more accurate than time domain methods
  - However, mathematically they are more complex
- Finite element method (FEM) is the most computational tool used in all areas of engineering including electromagnetic simulation (e.g. HFSS, CST and COMSOL)
  - It solves partial differential equations by discretisation into *finite elements*

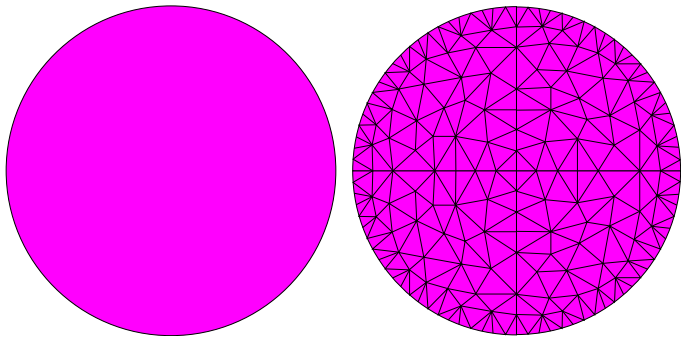


Figure 2: Typical mesh in FEM

- We observe the numerical time evolution of electromagnetic fields or currents through a region of space
- They are more intuitive as we can *visualise* wave phenomena
- We can obtain all the frequency information in one run through the Fourier transformation
- Simplicity comes at the cost of problems such as numerical stability and dispersion
- However, we will look into the finite difference time domain (FDTD) method deeply

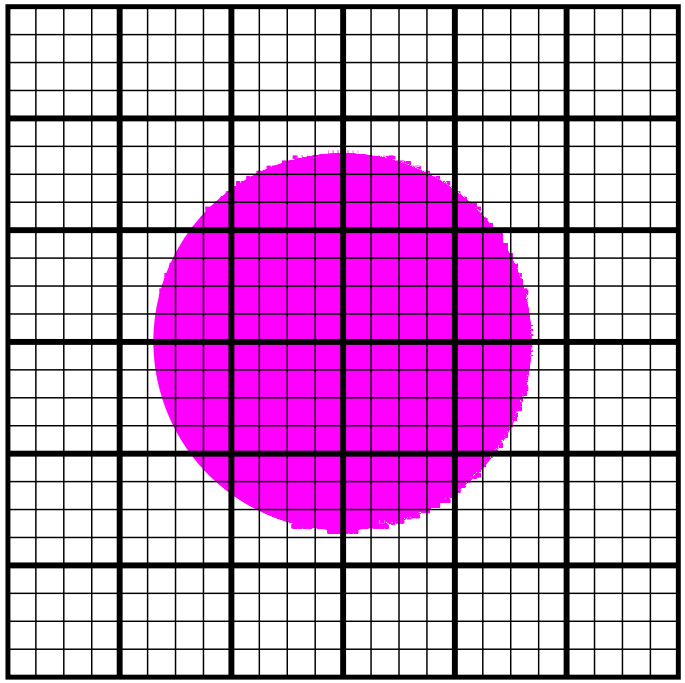
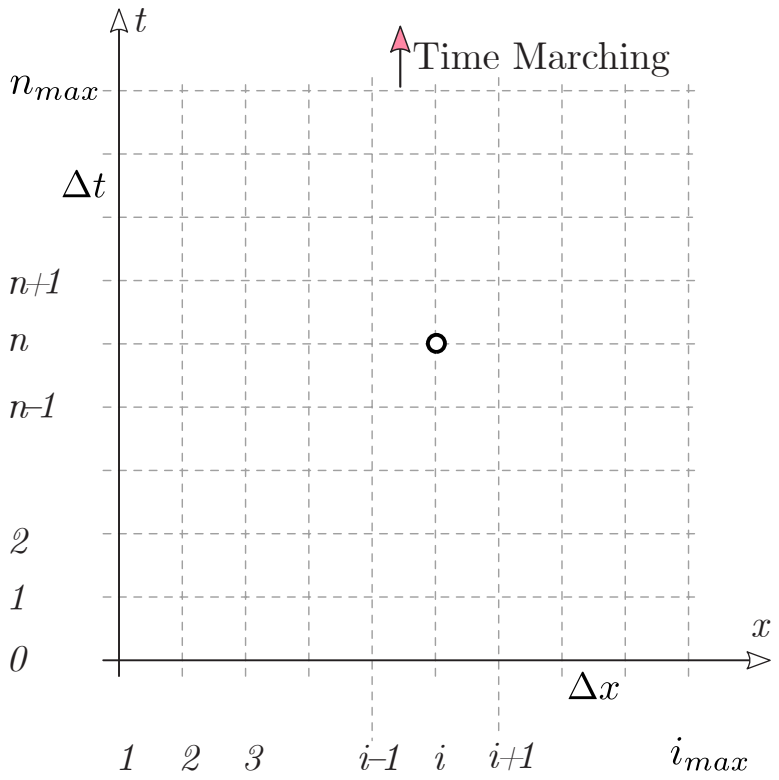


Figure 3: Typical mesh in FDTD

SOLVING DIFFERENTIAL EQUATIONS

- The goal is to solve differential equations *numerically* given some initial and boundary conditions
- Maxwell's equations has operators such as curl $\boldsymbol{\nabla}\times$, divergence $\boldsymbol{\nabla}\cdot$, and Laplacian ∇^2
- Lets first express the derivative term as a difference formula.

- We aim to transform a Calculus problem into a linear algebra one
- In other words, through discretisation, we move from the continuous domain to the discrete domain
- A partial differential equation is thus converted into a difference equation



Consider the Taylor series expansions of the function $f(x)$ expanded about the point x_0 with an offset of $\pm\Delta/2$:

$$f(x_0 + \frac{\Delta}{2}) = f(x_0) + \frac{\Delta}{2}f'(x_0) + \frac{1}{2!}\left(\frac{\Delta}{2}\right)^2 f''(x_0) + \frac{1}{3!}\left(\frac{\Delta}{2}\right)^3 f'''(x_0) + \ldots,$$

$$f(x_0 - \frac{\Delta}{2}) = f(x_0) - \frac{\Delta}{2}f'(x_0) + \frac{1}{2!}\left(\frac{\Delta}{2}\right)^2 f''(x_0) - \frac{1}{3!}\left(\frac{\Delta}{2}\right)^3 f'''(x_0) + \ldots$$

here the primes indicate differentiation. Subtracting the second equation from the first yields,

$$f\left(x_0 + \frac{\Delta}{2}\right) - f\left(x_0 - \frac{\Delta}{2}\right) = \Delta f'(x_0) + \frac{2}{3!}\left(\frac{\Delta}{2}\right)^3 f'''(x_0) + \ldots \qquad (1)$$

Moving on and dividing Eq. 1 by Δ we get,

$$\frac{f\left(x_0+\frac{\Delta}{2}\right)-f\left(x_0-\frac{\Delta}{2}\right)}{\Delta}=f'(x_0)+\frac{1}{3!}\frac{\Delta^2}{2^2}f'''(x_0)+\dots$$

The term on the left is the derivative term of the $f(x)$ at point x_0 plus an approximation term, $O(\Delta^2)$:

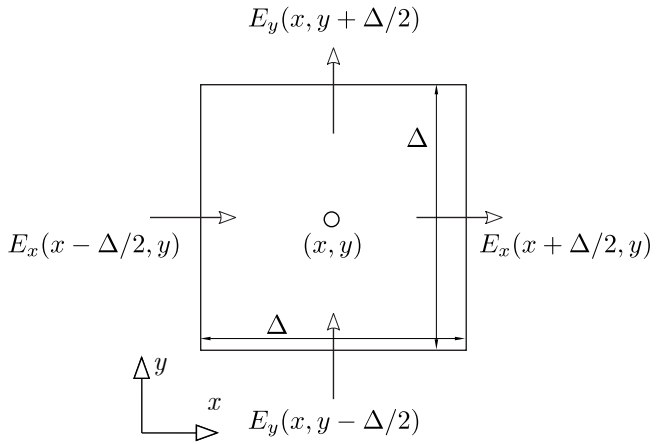
$$\frac{\mathrm{d}f}{\mathrm{d}x}|_{x=x_0}=\frac{f\left(x_0+\frac{\Delta}{2}\right)-f\left(x_0-\frac{\Delta}{2}\right)}{\Delta}+O(\Delta^2).$$

The central difference formula is thus:

$$\frac{\mathrm{d}f}{\mathrm{d}x}|_{x=x_0}\approx\frac{f\left(x_0+\frac{\Delta}{2}\right)-f\left(x_0-\frac{\Delta}{2}\right)}{\Delta}.$$

The divergence of a vector gives us a scalar field, and according to the Maxwell's equation:

$$\nabla \cdot E = \frac{\rho}{\varepsilon} = \frac{\partial E_x}{\partial x} + \frac{\partial E_y}{\partial y} + \frac{\partial E_z}{\partial z}$$

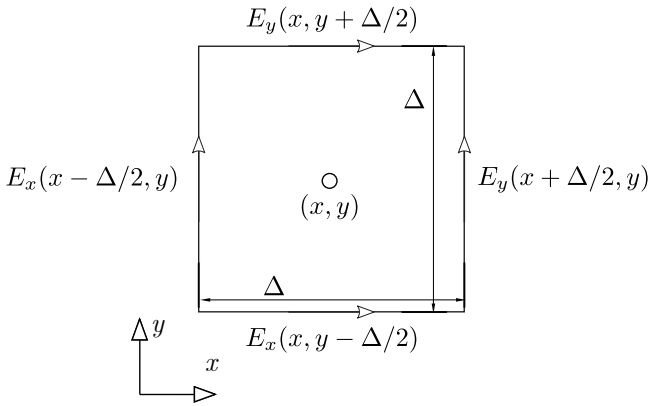


For a 2D case, the above expression can be written as:

$$\frac{\partial E_x}{\partial x} + \frac{\partial E_y}{\partial y} \approx \frac{(E_x(x + \Delta/2, y) - E_x(x - \Delta/2, y) + E_y(x, y + \Delta/2) - E_y(x, y - \Delta/2))}{\Delta}$$

Likewise, we use the curl $\nabla\times$ operator in Maxwell's equations:

$$\nabla \times E = \begin{bmatrix} \widehat{\mathbf{x}} & \widehat{\mathbf{y}} & \widehat{\mathbf{z}} \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\ E_x & E_y & E_z \end{bmatrix}$$



For a 2D case, the above expression can be written as:

$$\frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y} \approx \frac{(E_y(x + \Delta/2, y) - E_y(x - \Delta/2, y) - E_x(x, y + \Delta/2) + E_x(x, y - \Delta/2))}{\Delta}$$

The wave equation possesses the ∇^2 operator:

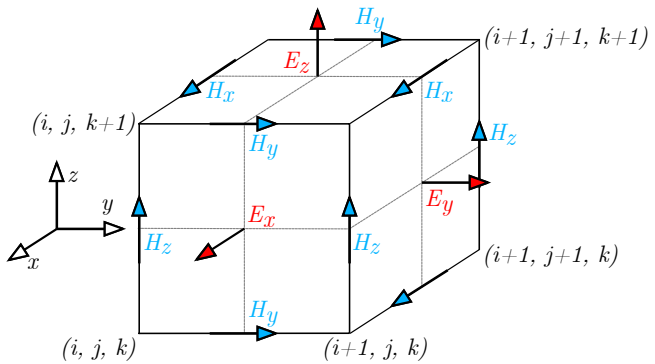
$$\nabla^2\phi = \frac{\partial^2\phi_x}{\partial x^2} + \frac{\partial^2\phi_y}{\partial y^2} + \frac{\partial^2\phi_z}{\partial z^2}$$

Using the Taylor Series expansion, we can write the second derivate as:

$$\frac{\mathrm{d}^2 f}{\mathrm{d}x^2}\Big|_{x=x0} \approx \frac{f(x_0+\Delta) - 2f(x_0) + f(x_0-\Delta)}{\Delta^2}$$

A point in space in a uniform rectangular lattice can be denoted as $(i\Delta x, j\Delta y, k\Delta z)$. Any function can then be expressed in terms of space and time

$$u(i\Delta x, j\Delta y, k\Delta z, n\Delta t) = u^n_{i,j,k}$$



- We are going to compute *only one* component at a given point
- There are no field components at the lattice edges
- Each component is similarly spaced.

- Developed in 1966 by Kane Yee
- The algorithm employs second-order ($O(\Delta^2)$) central differences.

1. Replace all the derivatives in Ampere's and Faraday's laws with finite differences.
   1.1 Discretize space and time so that the electric and magnetic fields are staggered in both space and time.
2. Evaluate the magnetic fields one time-step into the future ($n + 1$)
3. Evaluate the electric fields into the future ($n + 1$)
4. Repeat Steps 3 and 4 until all the fields have been computed.

ONLINE NOTEBOOK

- Google Colab notebook running a Python example
- We focus on 1D FDTD first.
- Alternatively, download the Python repository from:

Let's write the time-dependent Maxwell's equations that have the curl operations:

$$\frac{\partial \vec{\mathbf{E}}}{\partial t} = \frac{1}{\varepsilon_0} \boldsymbol{\nabla} \times \vec{\mathbf{H}}$$
$$\frac{\partial \vec{\mathbf{H}}}{\partial t} = -\frac{1}{\mu_0} \boldsymbol{\nabla} \times \vec{\mathbf{E}}$$

For one-dimensional case, we have:

$$\frac{\partial E_z}{\partial t} = \frac{1}{\varepsilon_0} \frac{\partial H_y}{\partial x}$$
$$\frac{\partial H_y}{\partial t} = -\frac{1}{\mu_0} \frac{\partial E_z}{\partial x}$$

We will be using the finite difference notation to express the fields and derivatives (differences)

$$E_z(x,t) = E_z\left(i\Delta_x, n\Delta_t\right) = E_z^n[i]$$
$$H_y(x,t) = H_y\left(i\Delta_x, n\Delta_t\right) = H_y^n[i]$$

Here Δ_x and Δ_t are offsets in space and time. Indices i and n represent steps in space and time.

The 1D Maxwell's (Faraday's) equations then become:

$$\mu \frac{\partial H_y}{\partial t}\bigg|_{(i+1/2)\Delta_x, n\Delta_t} = \frac{\partial E_z}{\partial x}\bigg|_{(i+1/2)\Delta_x, n\Delta_t}$$

$$\mu \frac{H_y^{n+\frac{1}{2}}\left[i+\frac{1}{2}\right] - H_y^{n-\frac{1}{2}}\left[i+\frac{1}{2}\right]}{\Delta_t} = \frac{E_z^n[i+1] - E_z^n[i]}{\Delta_x}$$

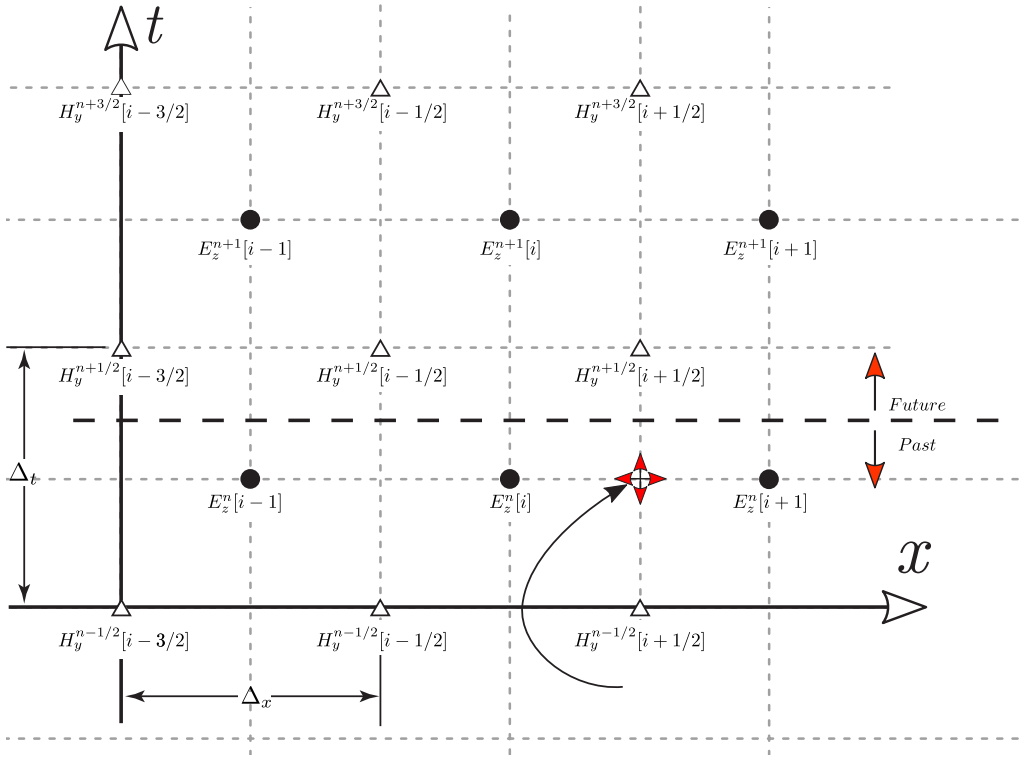


Figure 4: The interleaved E_z and H_y points in space and time.

$$H_y^{n+\frac{1}{2}}\left[i+\frac{1}{2}\right] = H_y^{n-\frac{1}{2}}\left[i+\frac{1}{2}\right] + \frac{\Delta_t}{\mu\Delta_x}\left(E_z^n\left[i+1\right] - E_z^n\left[i\right]\right). \qquad (2)$$

Using similar techniques, we get the update equation for the electric field using the Ampere's law:

$$E_z^{n+1}[i] = E_z^n[i] + \frac{\Delta_t}{\varepsilon \Delta_x}\left(H_y^{n+\frac{1}{2}}\left[i+\frac{1}{2}\right] - H_y^{n+\frac{1}{2}}\left[i-\frac{1}{2}\right]\right). \tag{3}$$

- We have the update coefficients $\frac{\Delta_t}{\varepsilon \Delta_x}$ and $\frac{\Delta_t}{\mu \Delta_x}$ in the electric and magnetic fields expressions.
- These coefficients determine how far the energy propagates in a single step in time and space.
- Knowing $c = 1/\sqrt{\varepsilon_0 \mu_0}$, the maximum distance energy can travel in *one* time step is $c\Delta_t$
- Courant number S_c is the ratio $c\Delta_t/\Delta_x$
  - It determines the stability of the simulation.

$$\frac{1}{\varepsilon}\frac{\Delta_t}{\Delta_x} = \frac{1}{\varepsilon_r \varepsilon_0}\frac{\sqrt{\varepsilon_0\mu_0}}{\sqrt{\varepsilon_0\mu_0}}\frac{\Delta_t}{\Delta_x} = \frac{\sqrt{\varepsilon_0\mu_0}}{\varepsilon_r\varepsilon_0}\frac{c\Delta_t}{\Delta_x} = \frac{1}{\varepsilon_r}\sqrt{\frac{\mu_0}{\varepsilon_0}}\frac{c\Delta_t}{\Delta_x} = \frac{\eta_0}{\varepsilon_r}\frac{c\Delta_t}{\Delta_x} = \frac{\eta_0}{\varepsilon_r}S_c$$

$$\frac{1}{\mu}\frac{\Delta_t}{\Delta_x} = \frac{1}{\mu_r \mu_0}\frac{\sqrt{\varepsilon_0\mu_0}}{\sqrt{\varepsilon_0\mu_0}}\frac{\Delta_t}{\Delta_x} = \frac{\sqrt{\varepsilon_0\mu_0}}{\mu_r\mu_0}\frac{c\Delta_t}{\Delta_x} = \frac{1}{\mu_r}\sqrt{\frac{\varepsilon_0}{\mu_0}}\frac{c\Delta_t}{\Delta_x} = \frac{1}{\mu_r\eta_0}\frac{c\Delta_t}{\Delta_x} = \frac{1}{\mu_r\eta_0}S_c$$

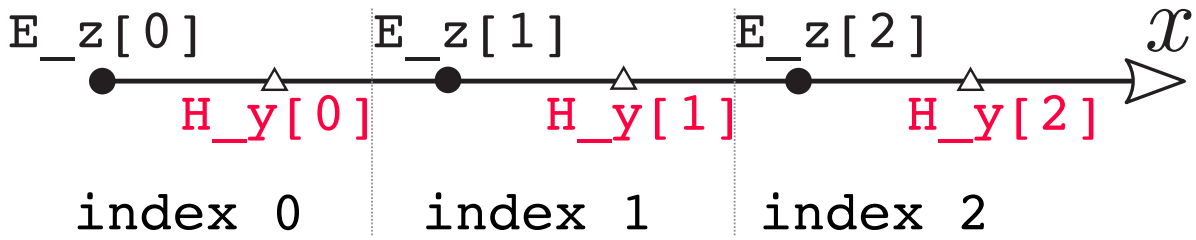


Figure 5: Array variable definition for 1D FDTD.

The update equations for a computer program therefore are:

```
hy[i] = hy[i] + (ez[i + 1] - ez[i]) / imp0
```

```
ez[i] = ez[i] + (hy[i] - hy[i - 1]) * imp0
```

Here `imp0` is the characteristic impedance, and $S_c = 1$ is assumed 1. Note the magnetic field is calculated first and then the electric field. We place the expressions above in a loop.

- The points at the boundary need to be dealt with differently. As an example, for `ez[0]`, we dont have `hy[-1]`.
  - Similarly, for `hy[MAX-1]`, there is no `ez[MAX]`
  - We can apply absorbing boundary conditions
- Only having one impedance results in a homogeneous medium
  - We can assign each node a ε and μ
- Right now we haven't talked about introducing energy into the system
  - We can create a hard-wired source

A SIMPLE PYTHON 1D FDTD SIMULATION

```
# do time stepping
for n in range(maxTime):

# update magnetic field
for i in range(SIZE-1):
hy[i] = hy[i] + (ez[i + 1] - ez[i]) / imp0

# update electric field
for i in range(SIZE):
ez[i] = ez[i] + (hy[i] - hy[i - 1]) * imp0

# Additive Gaussian source node */
if n < sourceSigma:
    ez[0] = math.exp(-((n - 1*sourcePeakTime)/)**2 / sourceSigma)
else:
    ## PEC Boundaries
    ez[0] = 0.0
    ez[(SIZE - 1)] = 0.0

print(ez[sensorLocation])
#done with time stepping loop
```

To simulate infinite space, we use special types of boundary conditions called absorbing boundary conditions (ABCs).
ABCs are approximate and introduce dispersion especially in higher dimensions
Simple ABC's are:

```
ez[0] = ez[1]
hy[MAX-1] = hy[MAX-2]
```

Figure 6: F-16 inside an RF anechoic test chamber.

- In FDTD, we can either use *hard-wired* sources or *additive sources*.
- The former is easier to implement but suffers from severe discontinuity on the point of generation
- The latter just introduces a current term in the electric field update equation.

```
if n < sourceSigma:
    ez[0] = math.exp(-(n - sourcePeakTime)**2 / sourceSigma)
else:
    ez[0] = 0.0
```

Hardwired source

```
ez[50] += math.exp(-(n - sourcePeakTime)**2 / sourceSigma)
```

Additive Source

Recall we have the material coefficients in the electric and magnetic field update equations:

$$H_y^{n+\frac{1}{2}}\left[i+\frac{1}{2}\right] = H_y^{n-\frac{1}{2}}\left[i+\frac{1}{2}\right] + \frac{\Delta_t}{\mu\Delta_x}\left(E_z^n\left[i+1\right] - E_z^n\left[i\right]\right)$$

$$E_z^{n+1}\left[i\right] = E_z^n\left[i\right] + \frac{\Delta_t}{\varepsilon\Delta_x}\left(H_y^{n+\frac{1}{2}}\left[i+\frac{1}{2}\right] - H_y^{n+\frac{1}{2}}\left[i-\frac{1}{2}\right]\right)$$

We can therefore define ε and μ at any point in the FDTD simulation:

```
hy[i] = hy[i] + (ez[i + 1] - ez[i]) / imp0 / muR[i]
ez[i] = ez[i] + (hy[i] - hy[i - 1]) * imp0 / epsR[i]
```

Chapters 2 & 3 — Understanding the FDTD Method[1]

[1] John Schneider. *Understanding the FDTD Method*. URL: https://eecs.wsu.edu/~schneidj/ufdtd/.